

Output Error Estimates and Mesh Refinement in Aerodynamic Shape Optimization

Marian Nemec*

Science & Technology Corp., Moffett Field, CA 94035, USA

Michael J. Aftosmis†

NASA Ames Research Center, Moffett Field, CA 94035, USA

We investigate the control of discretization error in gradient-based aerodynamic shape optimization through the use of adaptive mesh refinement. Shape optimization is usually performed on meshes with cell spacing and total mesh size determined *a priori*. In this work, we adapt the mesh at each design iteration to improve accuracy and reduce optimization setup time by eliminating the need to hand-craft a mesh that is suitable for all design iterations. The approach makes dual use of the adjoint method – first in the computation of the objective function gradient and second in the estimation of discretization error. We study the convergence of error estimates and show vanishing and non-monotone behavior when the objective function involves quadratic forms. A companion functional is formulated to remove these deficiencies in a manner that does not increase the cost of the simulation. We explore the use of progressive optimization, where the depth of mesh refinement is systematically increased as the design improves. The approach is demonstrated on several model problems, including airfoil optimization and three-dimensional inverse design for sonic-boom control. These examples are used to examine issues important for the development of dynamic error control to improve automation and minimize cost.

I. Introduction

A persistent challenge in the application of Euler and Navier–Stokes simulations in numerical optimization is the accurate and efficient evaluation of the objective function, which is usually a functional of outputs such as lift, drag or moments. In an effort to address this issue, we examine the use of output error estimation^{1,2,3,4} with adaptive mesh refinement for the solution of aerodynamic shape optimization problems.

Output error estimates in simulation-based design offer several benefits. First, the ability to control objective-function discretization error improves confidence in the optimized designs. Put another way, design modifications should yield valid improvements in performance instead of false trends due to discretization error effects. Second, it eliminates the requirement of hand-crafting a sufficiently general mesh that is appropriate for all candidate designs. This is an important step toward making aerodynamic shape optimization tools available to the broader aerodynamics community. Third, direct control over the level of discretization error in the objective function allows the optimization procedure to begin on coarse meshes and progressively adjust the mesh as optimality is approached. This is similar to the familiar grid-sequencing startup of flow solvers and should significantly decrease the cost and turn-around time of the optimization.

Despite these benefits, there has been relatively little study of output error control in aerodynamic shape optimization. The current work is motivated by the ideas of Lu and Darmofal,⁵ Rannacher⁶ and Alauzet *et al.*⁷ on consistent approximations of functionals and gradients via the adjoint method.^{8,9,10} The basic idea is to ensure convergence of a sequence of discrete solutions to a local optimum of the continuous

*Research Scientist, Advanced Supercomputing Division, MS 258-5; marian.nemec@nasa.gov. Senior Member AIAA.

†Aerospace Engineer, Advanced Supercomputing Division, MS 258-5; michael.aftosmis@nasa.gov. Associate Fellow AIAA.

Copyright © 2013 by the American Institute of Aeronautics and Astronautics, Inc. The U.S. Government has a royalty-free license to exercise all rights under the copyright claimed herein for Governmental purposes. All other rights are reserved by the copyright owner.

design problem.^{11,12} This can be accomplished by controlling both the cell-density (mesh size) and the number of solver iterations in a way that minimizes the computational cost of the optimization. Dadone and Grossman¹³ propose a progressive optimization strategy with partially converged flow and adjoint solutions, but on uniformly refined grids. Recently, Hicken and Alonso¹⁴ developed a promising method for controlling discretization errors in gradients directly, but they require approximation of higher-order derivatives.

In previous work, we developed separate adjoint-based frameworks for output error estimation and for shape optimization. In error estimation, we extended the formulation of adjoint-weighted residuals to finite-volume discretizations on embedded-boundary Cartesian meshes.^{15,16,17} The results show that the new approach is accurate and efficient for routine use in practical settings, where the analysis of several thousand flight conditions and vehicle configurations may be required. In shape optimization, we applied the adjoint method to the computation of objective function gradients¹⁸ and developed a scalable framework¹⁹ to handle large conceptual design studies with complex geometry.^{20,21}

The purpose of this work is twofold. The first goal is to integrate our error estimation and mesh refinement procedure into the shape optimization framework, thus eliminating the time-consuming step of hand-crafting meshes and enabling the automatic construction of meshes tailored to each design iteration. The potential benefits include a much shorter optimization setup time and improved accuracy; however, the wall-clock time of each design iteration may increase relative to the traditional approach of using hand-crafted meshes of equivalent size. Hence, the second goal is to examine the turn-around time. We investigate the convergence of error estimates for various objective functions because these are key to evaluating design improvement. Thereafter, a progressive strategy of systematically increasing the depth of mesh refinement is used to examine issues hampering automation and efficiency of the optimization procedure. While our hope is to apply error control to practical shape optimization problems in an engineering setting, this paper focuses on understanding the behavior of this approach by investigating the performance of error control on relatively simple aerodynamic examples.

In Section II we define the optimization problem and briefly describe the optimizer. In particular, we discuss modifications that improve convergence of gradient-based methods in non-smooth optimization. In Sections III and IV we give background on the evaluation of the objective function, the error estimate, and the shape optimization framework. The salient aspects of sensitivity analysis on embedded-boundary Cartesian meshes are highlighted and we discuss the integration of mesh adaptation into the optimization framework. The effectiveness of the new framework with adaptive meshing is then demonstrated on a two-dimensional problem in transonic flow. While we do not consider the control of discretization errors in the gradient, the accuracy of the gradient is examined on the adapted meshes. In Section V we discuss the suitability of design objective functions for driving mesh adaptation. We show that for objective functions in quadratic form, e.g. least-squares inverse design, the error estimates vanish at the optimal solution, effectively stopping mesh refinement. This difficulty is circumvented by introducing a companion functional from which we compute error estimates in the design objective. The proposed approach is evaluated on several airfoil aerodynamic optimizations. Our final test case is a three-dimensional off-body inverse design in supersonic flow with a progressive optimization strategy.

II. Optimization Problem

The aerodynamic optimization problem we consider in this work consists of determining values of design variables, X , that minimize a given objective function

$$\min_X \mathcal{J}(X, Q) \quad (1)$$

where \mathcal{J} represents a scalar objective function defined by an integral over a region of the flow domain, for example lift or drag, and $Q = [\rho, \rho u, \rho v, \rho w, \rho E]^T$ denotes the continuous flow variables^a. The flow variables satisfy the three-dimensional, steady-state Euler equations of a perfect gas within a feasible region of the design space Ω

$$\mathcal{F}(X, Q) = 0 \quad \forall X \in \Omega \quad (2)$$

which implicitly defines $Q = f(X)$.

The optimization problem is solved using the BFGS (Broyden-Fletcher-Goldfarb-Shanno) quasi-Newton method in conjunction with a backtracking, inexact line search.²² We follow the suggestions of Lewis and

^aThe notation assumes that X is a scalar to clearly distinguish between partial and total derivatives.

Overton^{23, 24} and use weak Wolfe conditions to terminate line searches. When the design space is not smooth, a standard line search with strong Wolfe conditions may stall and prematurely terminate the optimization because it may not find an acceptable steplength.

We use a discrete formulation for the computation of the gradient, $d\mathcal{J}/dX$, where the governing equations, Eqs. 1 and 2, are first discretized and then linearized. In the following sections, we briefly describe the methodology for evaluating the objective function and its gradient. Details are available in earlier work^{15, 16, 18} and for a complete description of the optimization framework see Ref. 19.

III. Evaluation of Objective Function with Adaptive Mesh Refinement

A. Flow Solution

Our goal is to compute a reliable approximation of the objective function $\mathcal{J}(X, Q)$ at each step of the optimization procedure. Let $\mathcal{J}_H(\mathbf{Q}_H)$ denote an approximation of the functional^b computed on an affordable Cartesian mesh with an average cell size H , which we call the *working* mesh. The vector $\mathbf{Q} = [\bar{Q}_1, \bar{Q}_2, \dots, \bar{Q}_N]^T$ is the discrete solution vector of the cell-averaged values for all N cells of the mesh and \mathcal{J}_H is the discrete operator used to evaluate the functional. The governing equations are discretized on a multilevel Cartesian mesh with embedded boundaries. The mesh consists of regular Cartesian hexahedra everywhere, except for a layer of body-intersecting cells, or *cut-cells*, adjacent to the boundaries, as illustrated in Fig. 1. The spatial discretization of Eq. 2 uses a cell-centered, second-order accurate finite volume method with a weak imposition of boundary conditions, resulting in a system of equations

$$\mathbf{R}_H(\mathbf{Q}_H) = 0 \quad (3)$$

The flux-vector splitting approach of van Leer is used. Steady-state flow solutions are obtained using a five-stage Runge–Kutta scheme with local time-stepping, multigrid, and a domain decomposition scheme for parallel computing.^{25, 26, 27}

B. Error Estimate

To approximate the objective function error $|\mathcal{J}(Q) - \mathcal{J}_H(\mathbf{Q}_H)|$, we consider isotropic refinement of the working mesh to obtain a finer mesh with average cell size h containing approximately $8N$ cells (in three dimensions) and estimate the error in $|\mathcal{J}_h(\mathbf{Q}_h) - \mathcal{J}_H(\mathbf{Q}_H)|$. Our approach follows the work of Venditti and Darmofal,²⁸ where Taylor series expansions about the working mesh solution give the following expression of the functional on the embedded mesh:

$$\mathcal{J}_h(\mathbf{Q}_h) \approx \mathcal{J}_h(\mathbf{Q}_h^H) - \underbrace{(\psi_h^H)^T \mathbf{R}_h(\mathbf{Q}_h^H)}_{\text{Adjoint Correction}} - \underbrace{(\psi_h - \psi_h^H)^T \mathbf{R}_h(\mathbf{Q}_h^H)}_{\text{Remaining Error}} \quad (4)$$

\mathbf{Q}_h^H and ψ_h^H denote a reconstruction of the flow and adjoint variables from the working mesh to the embedded mesh, and the adjoint variables satisfy the following linear system of equations^c

$$\left[\frac{\partial \mathbf{R}(\mathbf{Q}_H)}{\partial \mathbf{Q}_H} \right]^T \psi_H = \frac{\partial \mathcal{J}(\mathbf{Q}_H)}{\partial \mathbf{Q}_H}^T \quad (5)$$

In Eq. 4, the adjoint variables provide both a correction term that improves the accuracy of the functional on the working mesh and a remaining error term that is used to form an error-bound estimate. A difficulty with the remaining error term is that it depends on the solution of the adjoint equation on the embedded

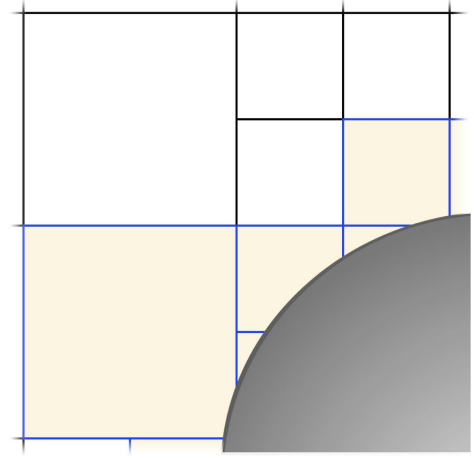


Figure 1. Multilevel Cartesian mesh in two-dimensions with a cut-cell boundary

^bIn this section we omit the functional dependence on X since the design variables are fixed during objective evaluations.

^cFor details on the implementation of the adjoint solver see Nemec and Aftosmis.²⁹

mesh, ψ_h . We approximate ψ_h with a triquadratic interpolant, ψ_Q , and ψ_h^H with a trilinear interpolant, ψ_L , from the adjoint solution on the working mesh.¹⁵ The remaining error term in Eq. 4 is used to estimate a bound on the local error in each cell k of the working mesh

$$e_k = \sum \left| (\psi_Q - \psi_L)^T \mathbf{R}_h(\mathbf{Q}_h^H) \right|_k \quad (6)$$

where the sum is performed over the children of each coarse cell and piecewise linear interpolation with limiters is used for the flow solution variable. An estimate of the net functional error E is the sum of the cell-wise error contributions to the functional

$$E = \sum_{k=0}^N e_k \quad (7)$$

The final expression for the variation of the corrected objective function due to discretization error is $\mathcal{J}_c \pm E$, where we assume no error cancellation in Eq. 6 and use the adjoint correction term in Eq. 4 to improve the accuracy of \mathcal{J}_H . Verification and validation studies presented in Refs. 16, 30, 31, 32 and 33 on both academic and practical problems indicate that this approach is the most effective way of estimating discretization errors in complex goal-oriented simulations.

C. Strategy for Mesh Adaptation

In previous work,^{15,16} we described a tolerance-driven strategy of mesh refinement based on a user-specified tolerance TOL for the objective function of interest. Starting from a coarse initial mesh, cells are flagged for refinement as indicated by the ratio of e_k to the allowable threshold TOL/ N . The solution is computed on the refined mesh and the adaptation cycles continue until the termination criterion $E \leq \text{TOL}$ is satisfied.

The basic idea is to equidistribute the cell-wise error as the adaptation advances. In other words, cells are refined such that each cell of the final mesh contributes roughly the same amount to the total error. Computational cost of the adaptation is reduced by using a “worst-cells-first” approach, where in the early cycles only the highest-error cells are flagged for refinement and as a result relatively few cells are added to the mesh. The mesh growth is increased as more cycles execute by adjusting the allowable threshold. This avoids the problem of generating too many cells early in the adaptive process and then carrying these cells through until the highest-error cells are addressed in the closing adaptation cycles. Note that one level of refinement is added per adaptation cycle and coarsening is not considered.

In recent practice, we have adopted a slightly modified strategy that is robust in difficult simulations with poor convergence. We rely less on the user-specified tolerance and instead specify the maximum number of adaptation cycles and a cell budget, i.e. the maximum number of cells that the adaptation is allowed to reach. In addition, the “worst-cells-first” approach is implemented by specifying a mesh-growth factor for each adaptation cycle. A mesh-growth factor of eight corresponds to uniform refinement, while a mesh growth factor of one indicates no refinement; a typical mesh-growth factor is two. Cells are sorted on their level of error and a fraction of the highest error cells is refined to meet the specified growth. This gives the user precise control over the number of cells in the mesh and therefore the allocation of computational resources. Additional cost savings are obtained by solving the adjoint equation and computing the error estimates up to and including the penultimate mesh but not on the finest mesh.

IV. Aerodynamic Shape Optimization

A. Gradient Computation

To compute the discrete gradient, $d\mathcal{J}_H/dX$, we recognize that a variation in X influences the computational mesh \mathbf{M} and the flow solution \mathbf{Q} . We rewrite Eq. 3 to explicitly include the design variables and the mesh, resulting in a system of equations

$$\mathbf{R}(X, \mathbf{M}, \mathbf{Q}) = 0 \quad (8)$$

where we omit the subscript H since all gradient computations are performed on the working mesh. The design variables that appear directly in Eq. 8 involve parameters that do not change the computational domain, such as the Mach number, angle of attack, and side-slip angle. The influence of shape design

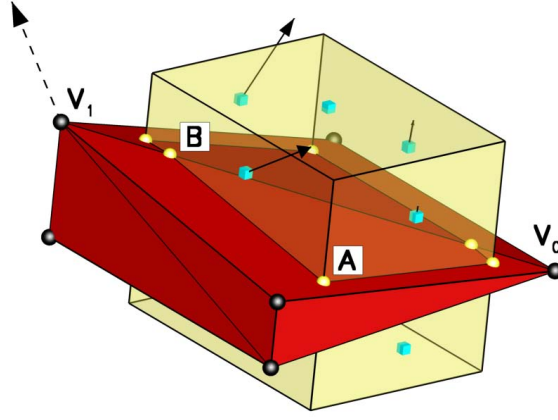


Figure 2. Sensitivity of face centroids (solid vectors) to perturbation of vertex V_1 (dashed vector)

variables on the residuals is implicit via the computational mesh

$$\mathbf{M} = f[\mathbf{T}(X)] \quad (9)$$

where \mathbf{T} denotes a triangulation of the wetted surface. The gradient is obtained by linearizing the objective function, $\mathcal{J}(X, \mathbf{M}, \mathbf{Q})$, and the residual equations, resulting in the following expression

$$\frac{d\mathcal{J}}{dX} = \frac{\partial \mathcal{J}}{\partial X} + \frac{\partial \mathcal{J}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{T}} \frac{\partial \mathbf{T}}{\partial X} - \psi^T \left(\frac{\partial \mathbf{R}}{\partial X} + \frac{\partial \mathbf{R}}{\partial \mathbf{M}} \frac{\partial \mathbf{M}}{\partial \mathbf{T}} \frac{\partial \mathbf{T}}{\partial X} \right) \quad (10)$$

where the adjoint vector ψ is given by Eq. 5.^d

The evaluation of Eq. 10 was presented previously in Ref. 18. We briefly review the evaluation of the partial derivative term $\frac{\partial \mathbf{M}}{\partial \mathbf{T}} \frac{\partial \mathbf{T}}{\partial X}$ in Eq. 10 because this term highlights the key differences between embedded-boundary and body-conforming approaches when considering shape sensitivities.

An infinitesimal perturbation of the boundary shape affects only the cut-cells. Consequently, the mesh-sensitivity term $\partial \mathbf{M} / \partial \mathbf{T}$, which contains the linearization of the Cartesian-face areas and centroids, volume centroids, and the wall normals and areas with respect to the surface triangulation, is non-zero only in these cells. The crux in the evaluation of $\partial \mathbf{M} / \partial \mathbf{T}$ is the linearization of the geometric constructors that define the intersection points between the surface triangulation and the Cartesian hexahedra. We explain the salient steps of the linearization using the example shown in Fig. 2, where a Cartesian hexahedron is split into two cut-cells by the surface triangulation. We require the linearization of the intersection points that lie on Cartesian edges, e.g. point A, and also those that lie on triangle edges, e.g. point B. Focusing on point B, its location along the triangle edge V_0V_1 is given by

$$B = V_0 + s(V_1 - V_0) \quad (11)$$

where s denotes the distance fraction of the face location relative to the vertices V_0 and V_1 . The linearization of this geometric constructor is given by

$$\frac{\partial B}{\partial X} = \frac{\partial V_0}{\partial X} + s \left(\frac{\partial V_1}{\partial X} - \frac{\partial V_0}{\partial X} \right) + (V_1 - V_0) \frac{\partial s}{\partial X} \quad (12)$$

A similar constructor is used for point A. Carrying this linearization through the various edge and face operations of the cut-cell construction gives the result shown in Fig. 2 for the position sensitivity of Cartesian face centroids given a perturbation at a single vertex (V_1).

Note that the “motion” of the face centroids is constrained to the plane of the face. An advantage of this formulation is that the dependence of the mesh sensitivities $\partial \mathbf{M} / \partial \mathbf{T}$ on the sensitivities of the surface triangulation ($\partial V_1 / \partial X$ in Eq. 12 and the term $\partial \mathbf{T} / \partial X$ in Eq. 10) is determined on-the-fly for each instance of the surface geometry. Put another way, there is no requirement for a one-to-one triangle mapping as the surface geometry changes. This allows a flexible interface for geometry control based on tools such as computer-aided design, as well as topology changes and refinement of the surface triangulation between design iterations.

^dEq. 10 represents a single component of the gradient vector for the general case of many design variables.

B. Framework Integration

The goal of the initial integration of the error estimation and mesh adaptation procedure into the optimization framework¹⁹ is to simply enable the use of adaptively refined meshes in each design iteration. This means that during an evaluation of the objective function, a fixed, user-specified number of mesh adaptation cycles is executed for each candidate design. The values of the objective and gradient from the finest mesh are passed to the optimizer. This is very similar to the traditional, fixed-mesh approach, but instead of building a single mesh at each design iteration we build a sequence of adapted meshes starting from a coarse baseline mesh. Since the mesh is custom-built for each design, we avoid meshing artifacts from previous designs. This is especially important in problems where changes to the geometry or freestream conditions generate very different flowfields, e.g. shocks and other nonsmooth features.

In problems where the objective function of the optimization problem can also be used to drive mesh adaptation, the adjoint variables associated with the the residual weights of the error estimate are the same as those associated with the objective gradient. Hence, we reuse the adjoint variables computed for error estimation directly in gradient evaluation, significantly reducing the cost of each design iteration because only one adjoint solution is required. It is important to note, however, that some objective functions are not appropriate for driving mesh adaptation. We return to this topic in Section V.

Overall, the integration results in essentially no changes to the framework architecture, except in the step of evaluating mesh sensitivities ($\partial \mathbf{M} / \partial \mathbf{T}$ in Eq. 10). The mesh linearization is implemented as a stand-alone code, independent of the mesh generator and the mesh adaptation module. An advantage of this implementation is that any mesh can be linearized, whether it comes from the mesh generator, as in the traditional, fixed-mesh approach, or the mesh adaptation module. This allows us to compute gradients for any mesh in the adaptation sequence. Furthermore, there are significant memory savings and speed improvements since only cut-cells are involved in the mesh linearization. For example, we gain a factor of three in memory reduction and a factor of four in speed relative to performing the mesh linearization within the mesh generator. By minimizing memory usage, we maximize the number of design variables that can be linearized in parallel.

C. Example

As a simple test of the integration of mesh adaptation within gradient-based optimization, we consider transonic flow around the NACA 0012 airfoil and seek to find an angle of attack to minimize the drag coefficient. This is a good model problem because small changes in the angle of attack can cause large changes in shock locations, requiring very different meshes as the optimization progresses. Many practical problems face similar issues, e.g. lift-constrained drag minimization for transport aircraft, where the wing shocks vary significantly as the optimization progresses. Our purpose is to study the accuracy of the objective function and gradient as the mesh is refined and demonstrate the effectiveness of this procedure for driving numerical optimization. The objective function is $\mathcal{J} = C_d$. The initial angle of attack is two degrees and the freestream Mach number is 0.8.

Figure 3 shows that the optimization converges in seven iterations. Drag (objective function) converges to just below 100 counts and the gradient is reduced by almost five orders of magnitude. The final angle of attack is almost zero (-0.001°). We start each design iteration from a coarse, essentially uniform mesh that contains 1,726 cells, with only 12 cells intersecting the airfoil. The near-field mesh around the airfoil is shown in Fig. 4. At each iteration, eight refinement passes are performed with a constant mesh growth factor to obtain final meshes of roughly 25,000 cells. Figure 5 shows the final meshes at selected design iterations. The refinement patterns clearly capture the design progression from a single, strong shock on the upper surface, to an intermediate, asymmetric two-shock system, to a final symmetric, weaker two-shock system at nearly zero angle of attack that minimizes drag.

Figure 6 shows the convergence of the error-estimation procedure at the first and third design iterations.

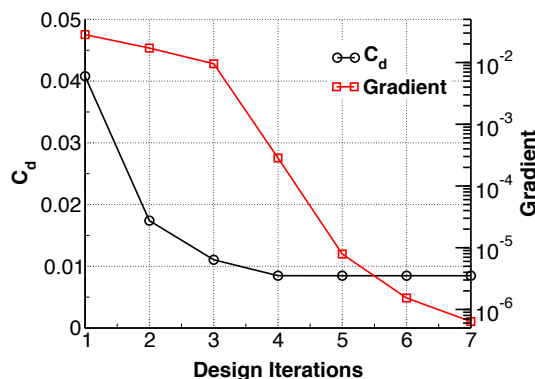


Figure 3. Convergence history for the transonic drag minimization problem

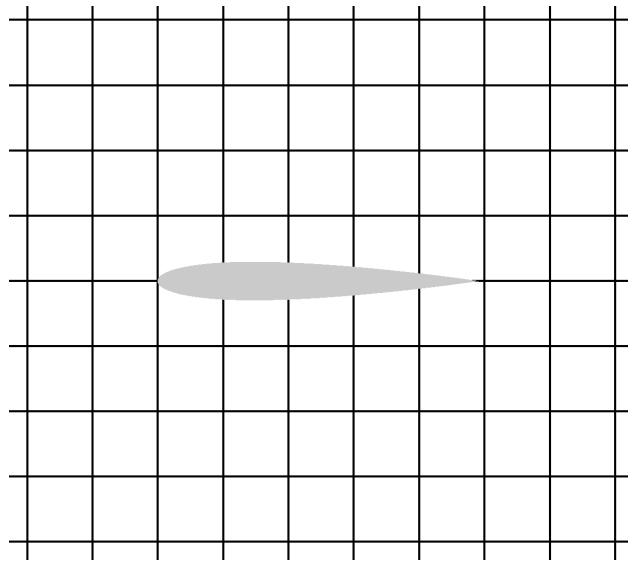


Figure 4. Near-field view of the initial mesh around the NACA 0012 airfoil. Mesh contains $\sim 1,700$ cells with just 12 cutcells

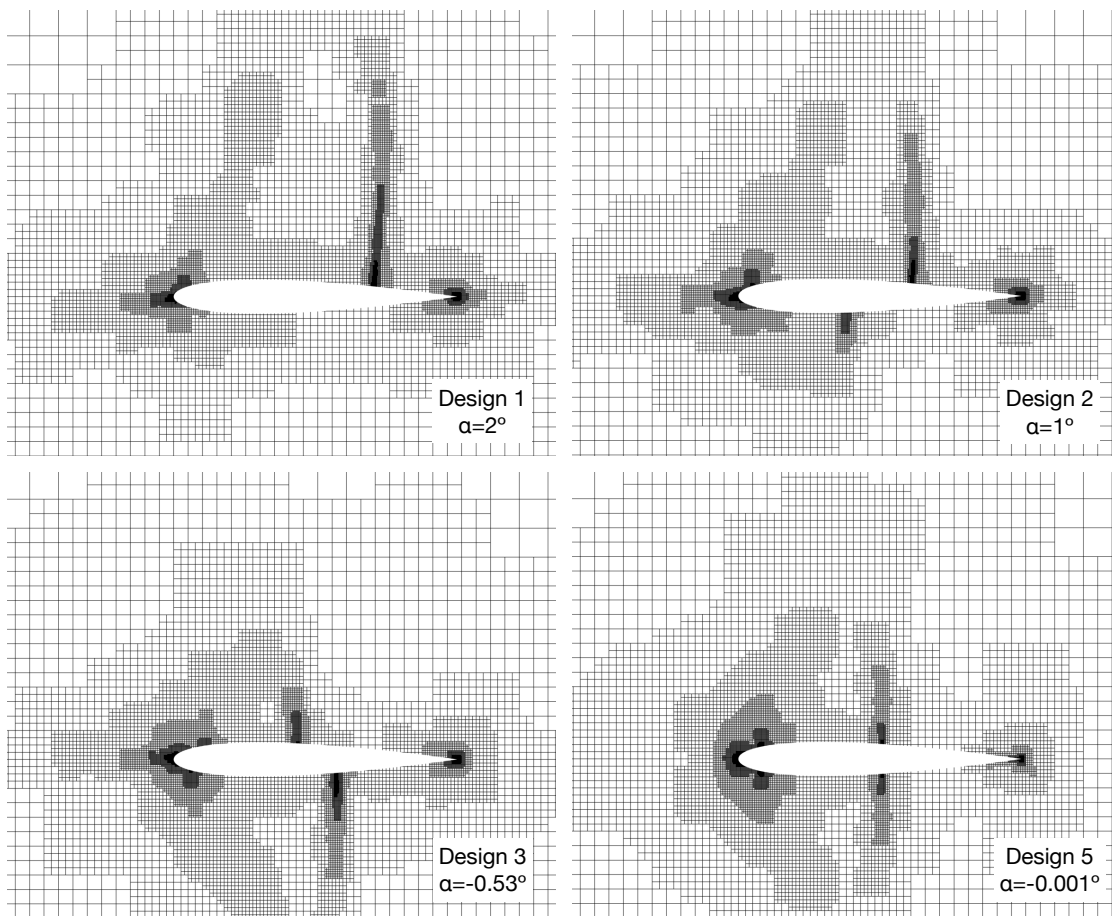


Figure 5. Near-field mesh evolution corresponding to angle of attack changes from initial to final designs. Meshes contain $\sim 25,000$ cells ($M_\infty = 0.8$)

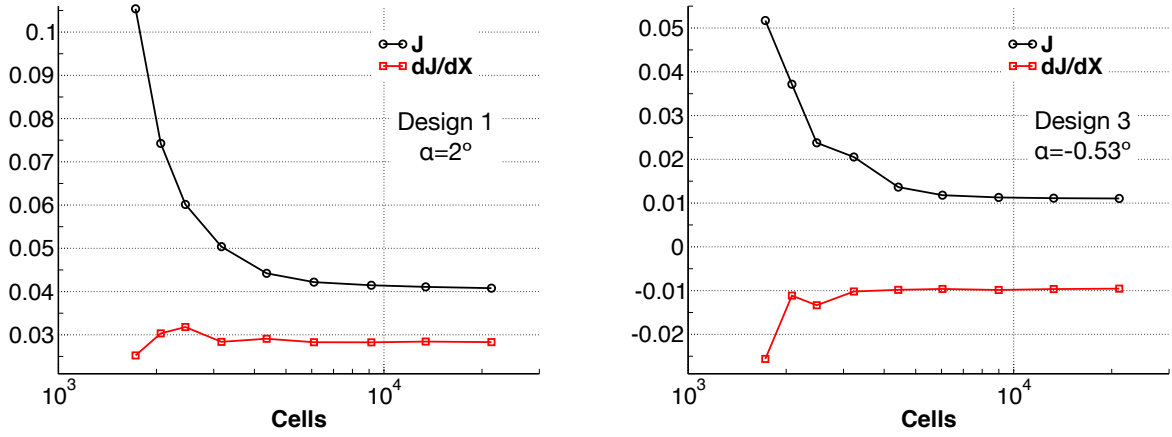


Figure 6. Mesh convergence of drag and its gradient at the first and third design iterations

Good convergence is observed for both the objective (C_d) and its gradient as the mesh is refined, with both values leveling out on a mesh with about 10,000 cells. The computation of the gradient on all meshes is not necessary in practice; we perform this computation here to examine the quality of the gradient on the coarse meshes. Figure 6 shows that the objective function is well converged over the last two mesh refinements. For this case, the gradient convergence rate is similar to the objective function and its sign is predicted correctly even on the initial mesh. Recall that the current error estimate is not targeting the gradient as an output of the simulation. This would require additional, higher-order terms.¹⁴ Convergence of the objective function error estimate is shown in Fig. 7. The error has decreased by over two orders of magnitude for each design and is smoothly decreasing in all adaptation cycles. The reduction in error as the optimization progresses is likely due to weaker shocks, which is reflected by the decreasing drag values (recall Fig. 3).

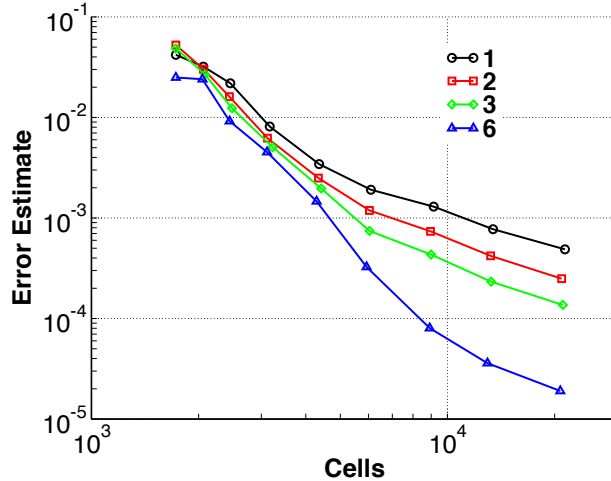


Figure 7. Mesh convergence of error estimate for drag at selected design iterations

V. Objectives in Quadratic Form

In practical optimization problems we frequently use objective functions that involve quadratic terms. For example, a term such as $(C_L - C_L^*)^2$ can be added to the objective function as a penalty to specify a target lift coefficient, C_L^* , for lift-constrained drag minimization problems. Inverse design problems use a similar formulation, where the objective function is specified as an integral of squared differences between the working variable and its target. When such objective functions are used in output error estimation, the computed error estimates vanish as the working variable approaches its target, e.g. as $C_L \rightarrow C_L^*$. This is because the right hand side of the adjoint equation, Eq. 5, is a derivative of the objective. Hence, the right hand side contains a linear term of the difference between the working variable and its target, e.g.

$2(C_L - C_L^*)$, which approaches zero as the working variable approaches the target. This causes the adjoint variables to vanish, yielding poor estimates of the correction and error terms in Eq. 4, and terminating mesh refinement.

An additional problem is that mesh convergence of the error estimate can be strongly non-monotone because the working variable may come closer to or drift farther from its target value as the mesh is refined during a design iteration. This may influence the quality of mesh refinement and consequently the accuracy of design analysis. Moreover, schemes that automatically adjust the mesh refinement level depending on the ratio of design improvement to the error estimate perform better when the error estimate converges smoothly. Similar observations have been made by Lu.³⁴ Rannacher⁶ discusses formulations for least-squares functionals in goal-oriented finite-element methods that avoid these difficulties.

To demonstrate, we modify the NACA 0012 airfoil problem of the previous subsection to find an angle of attack to match a target lift coefficient for an objective function in quadratic form: $\mathcal{J} = (C_l - 0.55)^2$. The target lift coefficient of 0.55 is achieved at roughly two degrees. The initial angle of attack is zero. We use the initial mesh shown in Fig. 4 and specify nine adaptation cycles for each design iteration. The final meshes are very similar to those shown in Fig. 5. The optimization converges in six design iterations as shown in Fig. 8.

The key plot is Fig. 9, which shows convergence of error estimates with mesh refinement at selected design iterations. The error estimates are well behaved over the first two design iterations, which is expected since the difference between the current lift and target lift is relatively large in early design iterations. The third design iteration shows non-monotone behavior. For this design, lift converges from above and happens to come very close to the target lift value on the seventh adaptation cycle. This causes a dip in the error curve down to $\approx 10^{-5}$. Thereafter, lift continues to converge to a value just below the target lift, which causes the error estimate to increase over the last two adaptation cycles because the difference $(C_l - C_l^*)$ is increasing on the last two meshes. On the final (sixth) design iteration, the error estimate vanishes on the finest mesh because the target lift is matched, falsely indicating that no further mesh refinement is required.

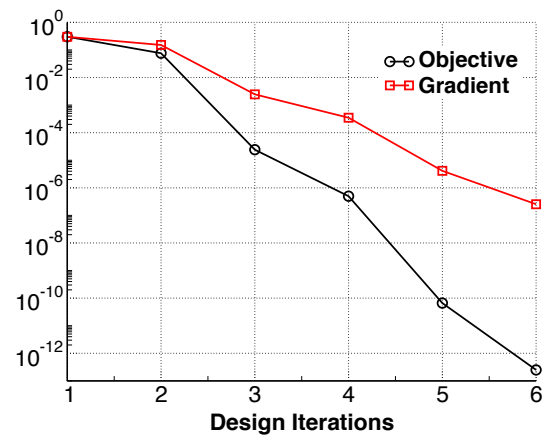


Figure 8. Convergence history for the transonic lift-matching optimization

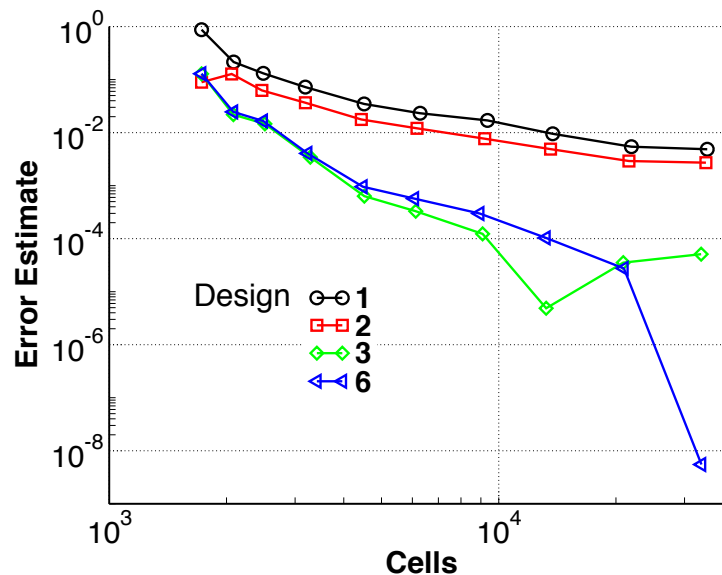


Figure 9. Mesh convergence of the error estimate for the lift-matching objective function

Recall that the drag error estimates converge smoothly for all designs with no numerical artifacts (see Fig. 7). This suggests a possible remedy for objective functions with quadratic terms. We propose to use a “companion functional” based on the outputs in the expression of the objective function for error-control and mesh refinement. We formulate the error-control functional such that the right hand side of the adjoint equation, Eq. 5, does not vanish when the optimal design is reached. Moreover, in ideal circumstances, the convergence of error-estimates of the companion functional is monotonic and this functional robustly drives mesh refinement to obtain reliable estimates of the objective function. To estimate errors in the objective, we propagate the error estimates from the companion functional through the objective function expression.

We note that it is quite easy to arrange the computations in each design iteration such that there is no additional cost due to the error-control functional. This is because in our strategy for mesh adaptation (recall Section III-C), we do not solve the adjoint equation and do not estimate the error on the finest mesh, instead we conservatively use the estimate from the penultimate mesh. Hence, the error-control functional is used on all but the finest mesh, where we swap out the error-control functional for the design objective.

We give a concrete example using the lift-matching optimization discussed above. For the objective function $\mathcal{J} = (C_l - 0.55)^2$, we use C_l as an error-control functional, \mathcal{J}_{EC} . The error in C_l , denoted by ε , is used to compute a conservative estimate of the objective function error as follows:

$$\mathcal{J} = ((C_l \pm \varepsilon) - 0.55)^2 \quad (13)$$

$$\leq (C_l - 0.55)^2 \pm \Delta \quad (14)$$

where

$$\Delta = |2(C_l - 0.55)\varepsilon| + \varepsilon^2 \quad (15)$$

When the difference between the current and target C_l is large, then the first term of the error expression dominates, but at optimality only the second term is non-zero.

We rerun the lift-matching optimization using $\mathcal{J}_{EC} = C_l$, i.e. we adapt the mesh and estimate errors in the objective function through use of C_l , and Eqs. 14 and 15. Figure 10 compares the objective function history between the original run, as shown in Fig. 8, and the new approach with the error-control functional. The new approach improves convergence of the design objective in almost all iterations^e. Lastly, in Fig. 11(a), we show mesh convergence of the error estimate obtained from Eq. 15 through the penultimate adaptation cycle. We observe no numerical artifacts – the behavior of the error estimates is now very similar to that in the drag minimization example shown in Fig. 7. In Fig. 11(b), we extend the computation to the finest mesh and compare the convergence of the error estimates for the final design of the optimization. The plot shows the original error estimate (E) that vanishes on the finest mesh (replot of blue-triangle line from Fig. 9), the error estimate in $\mathcal{J}_{EC} = C_l$ denoted by ε , and the corrected error estimate for the design objective computed via Eq. 15 denoted by Δ . The corrected error estimate decreases smoothly with mesh refinement.

Overall, the basic examples demonstrated so far indicate that output error estimates and adaptive meshing are effective in driving numerical optimization. The adapted meshes provide a reliable estimate of the objective function and gradient at each design iteration, resulting in good convergence of the optimizer. We also carefully verified the convergence of objective-function error estimates. These are important for not only reliable mesh adaptation but also dynamic error control during optimization when determining if a design improvement exceeds the level of discretization error on a given mesh. We now turn to a three-dimensional shape optimization problem, where these issues are examined further.

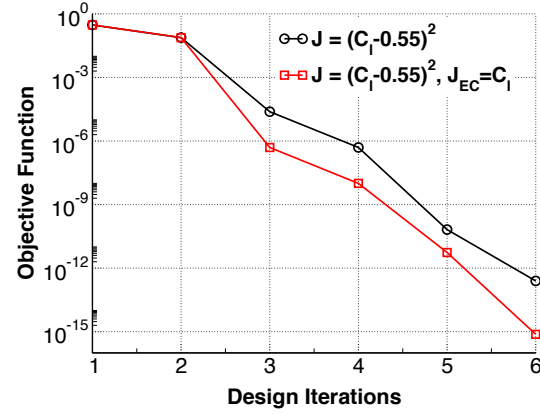
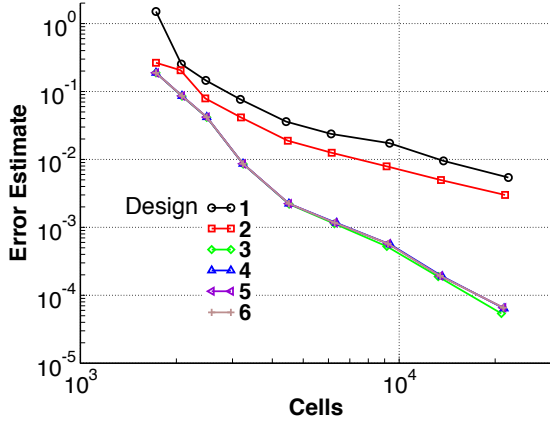
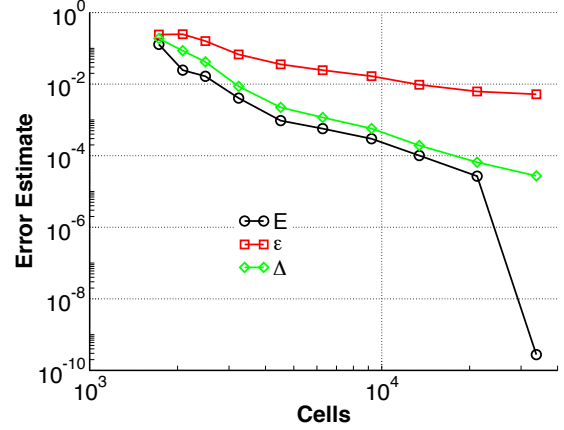


Figure 10. Objective function history for lift-matching optimization with and without error-control functional

^eImprovement in the first two iterations in Fig. 10 is not expected since the error estimate for the design objective is accurate when $C_l \approx C_l^*$.



(a) Error estimate for all design iterations on meshes up to and including the penultimate adaptation



(b) Convergence of original error (E), error in $\mathcal{J}_{EC}(\varepsilon)$, and corrected error estimate in design objective (Δ)

Figure 11. Mesh convergence of the objective function error estimate with error-control functional

VI. Results

We consider an inverse-design model problem with an off-body functional. Recent efforts in sonic-boom control^{20,35,36} have explored shaping a vehicle by prescribing a desired pressure signal in the flowfield some distance away. Figure 12 outlines the basic approach. As shown in the sketch, the domain is separated into a near-field region close to the body, and a far-field region primarily concerned with atmospheric propagation of the signal. In the near-field, three-dimensional effects are important in the composition of the pressure signal, while in the far-field the body can be considered axisymmetric, and the chief concerns are wave propagation and atmospheric signal distortion. The computational domain considers the near-field region, and a designer prescribes an off-body pressure signature with desirable characteristics. The optimizer seeks a shape that generates this signal by minimizing an objective function of the form

$$\mathcal{J} = \frac{1}{p_\infty^2} \int (p - p_{\text{target}})^2 dS \quad (16)$$

While Eq. 16 drives the design, this formulation is a clear example of the type of quadratic functional discussed earlier in Section V and is inappropriate for driving mesh adaptation. Specifically, the derivative of this objective contains the factor $(p - p_{\text{target}})$ and, as optimality is approached, the difference between the signature of the current design and that of the target will vanish causing the adjoint to vanish as well. We therefore drive mesh adaptation with an error-control functional that seeks to accurately predict the pressure along the sensor:

$$\mathcal{J}_{EC} = \frac{1}{p_\infty^2} \int (p - p_\infty)^2 dS \quad (17)$$

While the error-control functional is still in quadratic form, its derivative does not vanish because $p_{\text{target}} \neq p_\infty$ by construction. Moreover, the error estimate computed for Eq. 17 is a good approximation of error for Eq. 16 because they both have the same dependence on local pressure. Previous verification and validation studies have demonstrated detailed mesh convergence results for Eq. 17 and showed smoothly decreasing estimates of discretization error.^{20,30} Thus our solution strategy uses

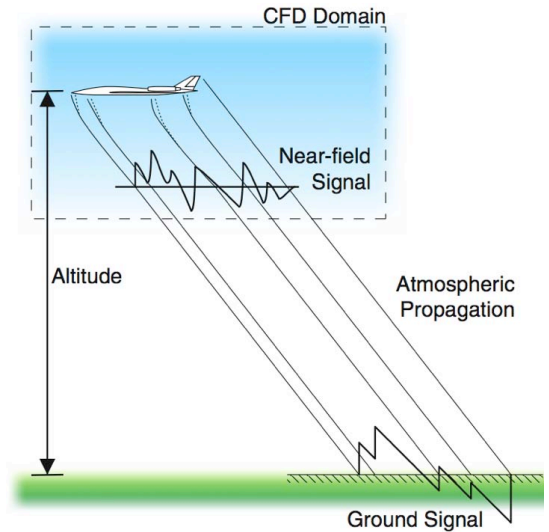


Figure 12. Illustration of domain decomposition for inverse-design in sonic-boom control

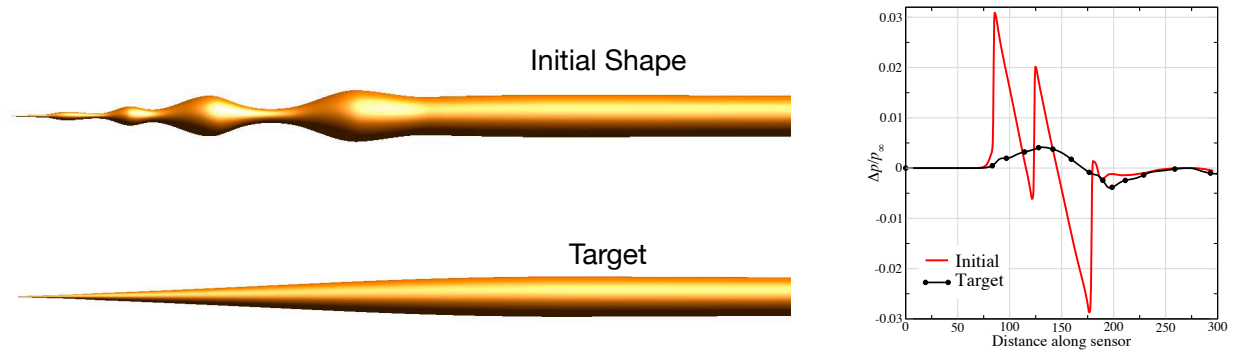


Figure 13. Left: Initial and target shapes used in the supersonic inverse-design problem. Ten design variables control the radius of the body at various streamwise stations. Right: Initial and target pressure signals at a distance of $h/L = 2$, $M_\infty = 1.5$, $\alpha = 0^\circ$

the target-matching functional, Eq. 16, to drive the shape optimization and Eq. 17 to control discretization error through mesh adaptation.

To demonstrate the methodology, we construct an inverse-design problem with an attainable solution and verify that the optimization can recover this solution from an arbitrary starting shape. We consider a slender, axisymmetric body in $M_\infty = 1.5$ supersonic flow at 0° incidence adopted from Wintzer and Kroo.²¹ Ten design variables control the radius of the body at various stations along its length. Figure 13 shows the shape that is used to generate the target pressure signal. Above the target shape, the figure shows the starting design which was obtained through gross perturbation of the design variables. On the right, the figure shows the pressure signals produced by these bodies measured at a distance of two body-lengths below the centerline ($h/L = 2$).

We consider two strategies for recovering the target shape. The first is a straightforward inverse design in which a fixed number of mesh-adaptation cycles are performed within each design iteration, similar to the airfoil demonstrations in Sections IV and V. The second example explores a basic progressive optimization strategy aimed at understanding issues associated with progressive design and giving insight into its potential for computational savings.

A. Shape Optimization with Fixed-Depth Adaptation

Figure 14 gives an overview of the optimization when using a fixed number of mesh adaptations for the evaluation of the objective function and gradient. The plot on the left shows convergence of both the objective function and gradient, while that on the right shows recovery of the target pressure signature by the final design.

Since the target is attainable, deep convergence of the objective function is expected, and the left frame of Figure 14 confirms this behavior. The abscissa of this plot measures cost by the total number of evaluations of the design objective and gradient (flow and adjoint solves on the finest mesh). Symbols on the lines indicate successful line searches. For this example with ten design variables, the objective function decreased by over six orders of magnitude in just over 50 iterations. Similar convergence behavior is seen in the design gradient. Optimization reduces the L_2 -norm of the design gradient by approximately half as many orders of magnitude as the objective function. Convergence is less smooth than for the objective function, but some noise is expected since the optimizer is performing inexact line searches with weak Wolfe conditions.

The adapted meshes in these simulations contained roughly 650,000 cells. To understand the computational cost of each design iteration, recall that each objective and gradient evaluation requires one flow and one adjoint solve, respectively, on the fine mesh. The cost for these solves is roughly equivalent to twice the cost of a single flow simulation. Additionally there is some cost for producing the adapted mesh. Adaptation is driven using the error-control functional \mathcal{J}_{EC} , which integrates pressure along the sensor via Eq. 17. The adaptation procedure constructs the mesh to minimize discretization error in this integral. The role of the adjoint solve during meshing, however, is limited to guiding mesh refinement, and this secondary adjoint need not be solved on the fine mesh. With mesh growth factors of roughly two at each adaptation cycle, the total cost for a design iteration is slightly over three times that of a single flow solve on the fine mesh.

Figure 15 shows the adaptive mesh on the initial and final design iterations. While both meshes stem from the same background mesh of 11,000 cells, after seven adaptation cycles, they have grown to $\sim 650k$ cells and

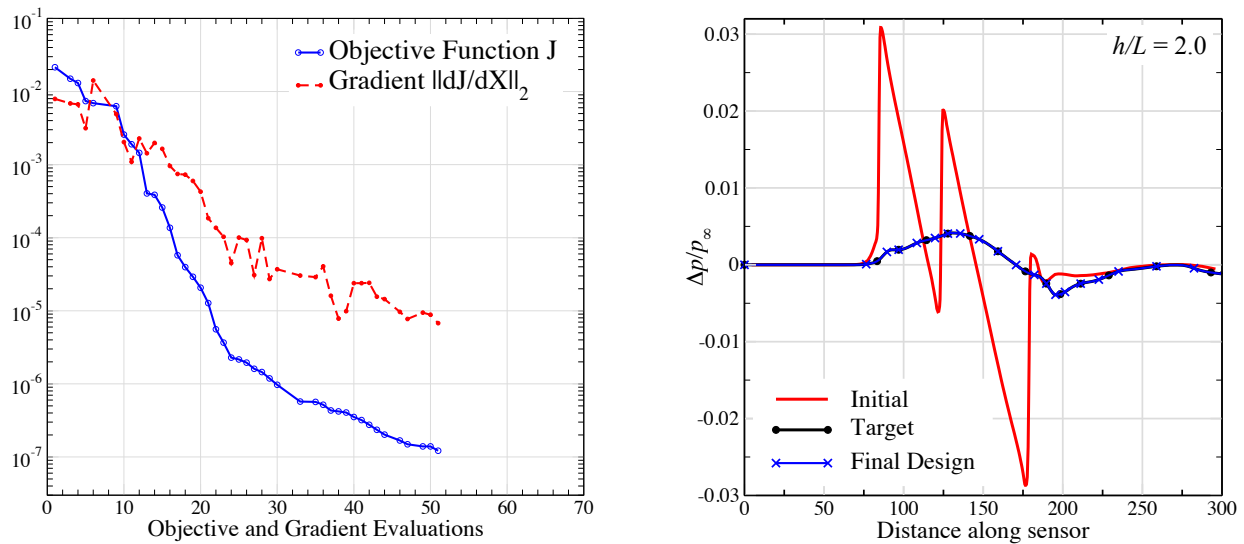


Figure 14. Left: Convergence history of the objective function and gradient for the supersonic inverse-design problem. Right: Pressure signature of the initial design, final design and optimization target measured at a distance of two body lengths ($h/L = 2.0$) off the centerline. Symbols are used to help distinguish the final design and target

are very different. As outlined in Subsection III-C, mesh adaptation is governed by a growth-based strategy which addresses the worst-cells-first at each adaptation cycle. Since the growth schedule and adaptation depth are fixed, the final mesh size is roughly constant throughout the entire design. Under this premise, the net effect of adaptation is to re-grade the cell densities within each design, tailoring the mesh to produce accurate pressures along the sensor with a fixed-cell budget. The snapshots of the mesh in Fig. 15, from the first and last design iterations, show the clear advantage of this approach. While the initial mesh focuses on accurately propagating the shocks and expansions emanating from the body, the final mesh is chiefly concerned with propagating a much smoother flowfield and dedicates more cells to accurately resolving the smooth, non-linear, near-body flow and accurately integrating along the sensor.

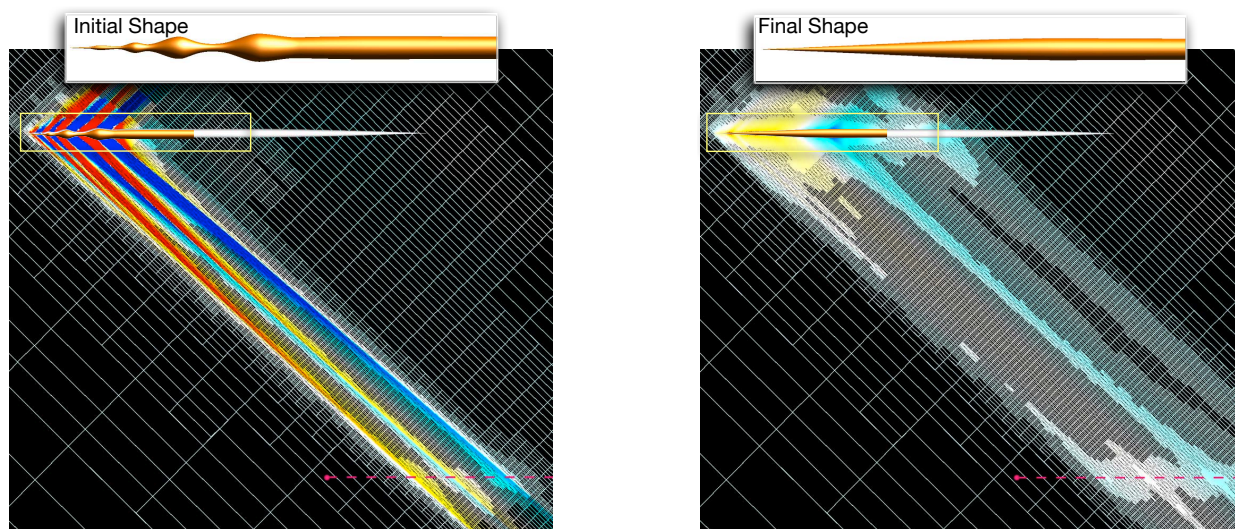


Figure 15. Adapted meshes on the initial and final design for 3D inverse design example with an off-body objective function using a fixed-depth adaptation strategy. All meshes contain $\sim 650k$ cells and are constructed with 7 levels of adaptive refinement. $M_\infty = 1.5$, $\alpha = 0^\circ$

B. Progressive Optimization

In the preceding example, all design iterations use the same level of adaptive mesh refinement. An obvious extension of this fixed-depth strategy is to adjust the level of refinement so that the mesh is progressively refined as the design advances, reducing the required computational effort during early design iterations. We perform an exploration of a progressive approach beginning with four levels of refinement and carrying through to seven levels of adaptation as we approach the target. The immediate goal is to uncover issues associated with the mechanics and cost of progressive optimization, and to use this understanding as we move toward development of an automated and efficient algorithm. From a cost standpoint, we wish to minimize the number of design iterations performed on the finest mesh. This exploration follows a simple strategy which lets coarse designs advance as far as possible and then uses those designs to initiate a new optimization seeded with the “best” design so far. At the outset, it is clear that this naive approach will not be as efficient as possible since we are essentially guaranteed to oversolve on the coarse grids. In other words, we do not use the ratio of design improvement to the error for determining transitions to the next level of refinement at this stage. In addition, the optimization resets the Hessian matrix with each progression in the refinement level and some subsequent design iterations are needed to rebuild this matrix.

Figure 16 summarizes the main results of this study. The frame at the left shows convergence of the design objective, Eq. 16, while at the right we see this convergence graphically as the off-body pressure recovers the target profile. Comparing these results with the fixed-depth example in Fig. 14 we see essentially the same depth of both objective convergence and target recovery. In the final sequence (7 adaptations), the optimization exited when updates to the design variables became smaller than a specified minimum. Reduction in the L_2 -norm of the gradient was similar to that observed in the earlier example (Fig. 14) and is omitted for clarity.

Tracing the convergence history, notice that each level of progressive refinement is accompanied by a small *increase* in the value of the objective function. This is an indication that \mathcal{J} is converging from below. We shall show shortly that this is not necessarily an indication of oversolving on a particular coarse mesh, but is rather caused by discretization error on the coarse mesh affecting the evaluation of the objective function. Given that this example evaluates a functional located two body-lengths away, it is not surprising that coarse meshes prematurely erode peaks and valleys in the pressure field as they propagate to the sensor.

Figure 17 gives more insight into the design evolution by presenting each stage of the optimization. This figure summarizes the 4 stages of progressive optimization by showing the initial and final designs for each stage along with their adaptive meshes (colored by pressure coefficient). Approximate mesh sizes are as follows: 4 Adaptations ~130k cells, 5 Adaptations ~230k cells, 6 Adaptations ~350k cells, 7 Adaptations ~650k cells. The right column shows the pressure signals measured at the sensor.

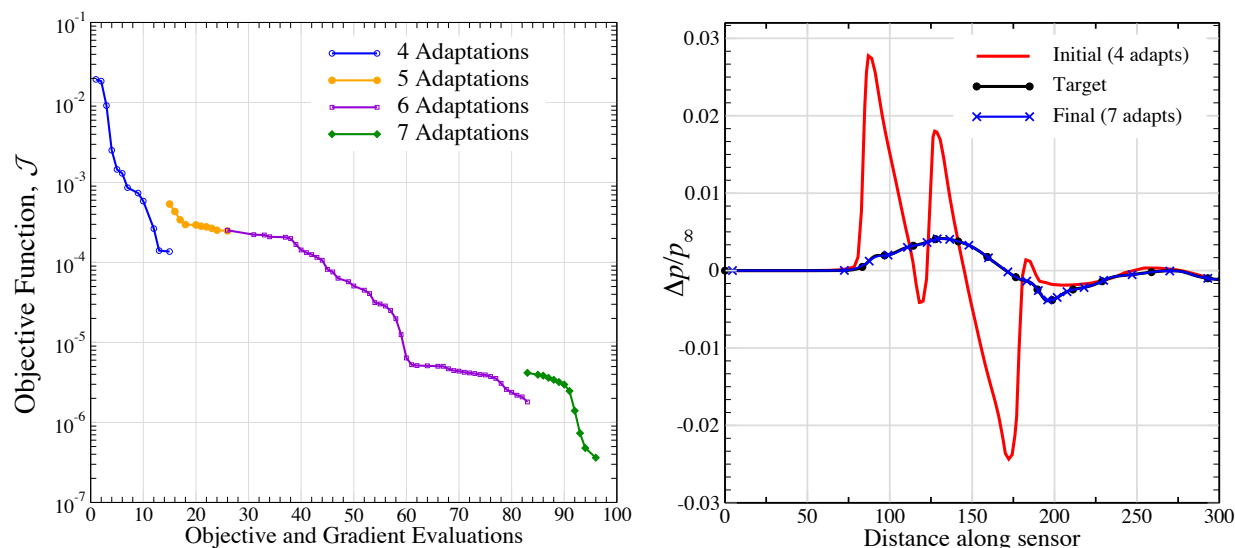
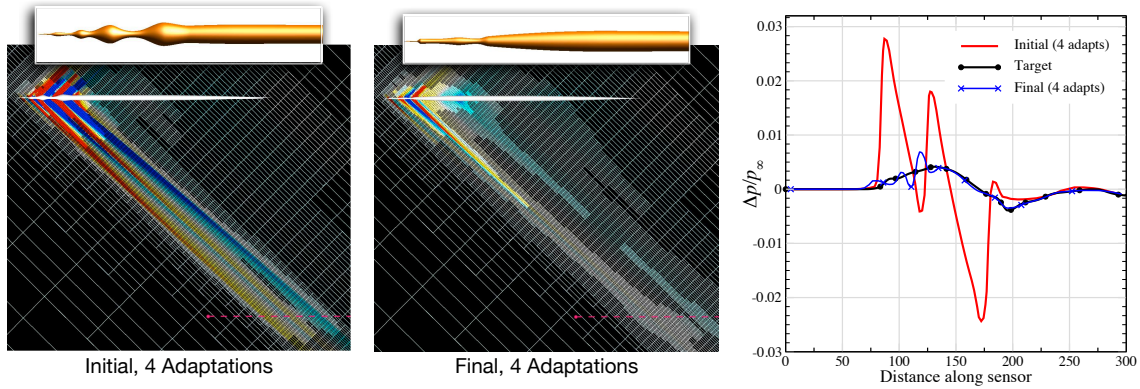
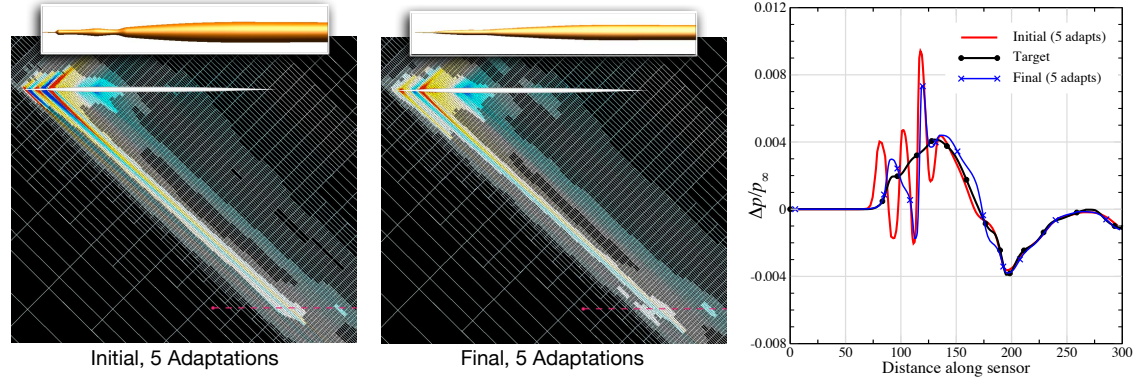


Figure 16. Left: Objective function convergence using progressive optimization from 4 to 7 levels of adaptive refinement. Right: Evolution of the off-body pressure signal used by the objective function, \mathcal{J} , in Eq. 16 from the initial design (4 levels of adaptation) to the final design (7 levels of adaptation)

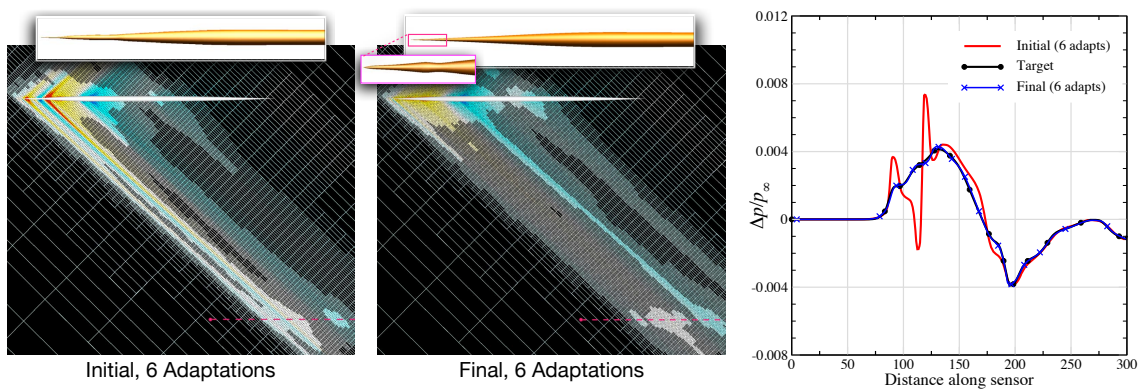
4 Adaptations



5 Adaptations



6 Adaptations



7 Adaptations

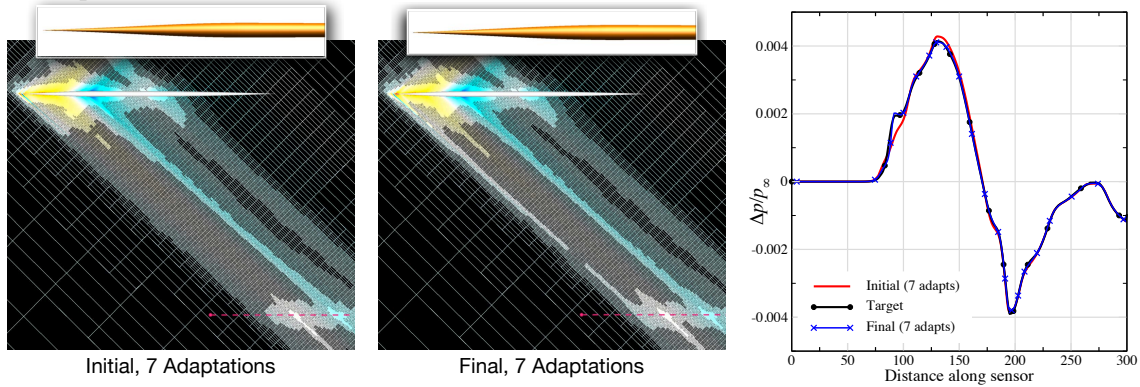


Figure 17. Summary of progressive optimization example for inverse design driven by an off-body objective function: 4 Adaptations ~130 k cells, 5 Adaptations ~230k cells, 6 Adaptations ~350k cells, 7 Adaptations ~650k cells. Meshes colored by pressure coefficient. Initial, final and target pressure signals are shown for each stage of optimization. $M_\infty = 1.5$, $\alpha = 0^\circ$, $h/L = 2$

The convergence history in Fig. 16 initially seems to indicate that most of the “work” was accomplished on the mesh with six levels of adaptation. While this is true for the computational effort, it is clear from the top row of Fig. 17 that the most radical reshaping of the body and cleanup of the pressure signal occurred during the first 15 iterations of the optimization on the coarsest adaptive mesh. This phase reduced the peak-to-peak signal by over a factor of five while reducing the objective function to less than 1% of its initial value. Despite mesh coarseness, the pressure signal of the initial design on the 4-level mesh compares well with that in Fig. 14 computed with 7 levels of adaptation. The first phase of adaptation stalled when the gradients pushed a design variable to violate its lower bound on the radius of the body near the pointed tip. This behavior is not surprising since the geometry at the first two design stations is substantially smaller than the cells in the coarse mesh.

The second phase of the objective convergence took place on a mesh with $\sim 230k$ cells and five levels of mesh adaptation. The convergence history in Fig. 16 shows a marked jump in the value of the objective when moving to this mesh, which is in part due to the increase in mesh resolution near the nose. Referring to Fig. 16, convergence on this mesh appears slow and reductions in the objective function are not dramatic. However, the second row of Fig. 17 gives more insight. The overall shape has progressed meaningfully. The body is smoother overall, and the very front of the signal has made substantial progress toward the objective at scales not resolvable on the previous mesh. This phase also terminated due to bounds violation in the under-resolved region near the tip.

With an additional level of adaptation, meshes in the third phase of design are $\sim 350k$ cells, with design iterations around half the cost of those in the fixed-depth example. The convergence history shows that this phase reduced the objective function by two orders of magnitude over the course of approximately 60 trial designs. In comparison with the convergence rate of the fixed-depth adaptation performed earlier, the design gradients on this mesh are less accurate, which may be a factor in the rate of design evolution. Nevertheless, the design progressed very well in this phase and the pressure signal of the final design is on top of the target to plotting accuracy. Upon closer inspection, the case is not so clear. As shown in the enlarged view in the third row of Fig. 17, the waviness at the body’s tip is still substantially different from the smooth tip of the target. In fact, it is clear that the optimizer is manipulating discretization errors to prematurely match the target signal with a false design. Re-evaluating the final design from the 6-level adaptive mesh to the finest mesh (7 adaptations) exposes the error on the previous mesh. From this initial design the optimizer recovers the actual target shape with about 12 objective and gradient evaluations.

This example brings up several points in understanding the potential for cost savings through progressive optimization. First and foremost is the degree to which allowable mesh growth impacts the cost arguments. In this example with growth factors around two, the design iterations on the penultimate mesh were about half as costly as those on the finest mesh. Iterations with the finest mesh started from a candidate design that was very close to the target, and still required about $1/3$ as many objective and gradient evaluations as the fixed-depth design. Under these circumstances, the mesh growth factor of two caps the potential payoff for progressive optimization at around a factor of two faster than the fixed-depth approach. Larger mesh growth factors permit higher savings and perhaps factors of four are achievable. However the example presented here shows that as long as shape parameters operate at scales not well resolved by the coarse meshes, a certain number of fine grid iterations are unavoidable. With this being the case, it is of interest to reduce the cost of the design iterations on the finest mesh as much as possible. For example, as the magnitude of design updates shrinks, it may be possible to re-use the adapted mesh for nearby designs on the finest mesh, thereby reducing or even eliminating the cost of the mesh adaptation procedure.

VII. Summary and Future Work

A framework for gradient-based aerodynamic shape optimization has been developed with the capability to perform output error estimation and adaptive mesh refinement in each design iteration. The examples demonstrate that this is a promising approach to enhance the accuracy, efficiency and automation of simulation-based design. They also highlight important points for designing an automated error-controlled method for optimization.

We showed that, without safeguards, the optimizer exploits discretization errors that lead to false designs. Moreover, there are significant costs associated with oversolving on coarse meshes. These not only include the useless design trials on the coarse mesh, but also the fine mesh design iterations required to “undo” the false progress. This penalty becomes more severe when oversolving occurs on mid-level meshes, requiring

additional iterations on ever finer meshes to correct the design. An additional penalty of oversolving is the contamination of the Hessian matrix. In our examples, the optimizer resets the approximate Hessian with each increase in adaptation depth. This is a conservative approach; however, it requires additional design iterations at the beginning of each phase to re-discover the design landscape.

An important metric for determining the acceptance of a design update, and whether to move to the next adaptation level, is the ratio of design improvement to the error estimate. We examined the convergence of objective-function error estimates in the presented examples. We found that for objective functions in quadratic form, a modification is required to avoid vanishing and non-monotone behavior of the estimates. There are several algorithms in the literature that automatically adjust the depth of mesh refinement depending on the design metric and attempt to safeguard Hessian updates as the design moves from mesh to mesh. Doing this efficiently for practical problems, without a priori knowledge of convergence of the objective function from below or above, and in conjunction with inexact line searches remains the subject of future work.

VIII. Acknowledgments

This work was supported by the NASA Ames Research Center contract NNA10DF26C, and the Subsonic Fixed Wing and Supersonics projects of NASA's Fundamental Aeronautics program. The authors gratefully acknowledge Mathias Wintzer (Analytical Mechanics Associates, Inc.) for providing the geometry modeler used in the inverse design problem.

References

- ¹Becker, R. and Rannacher, R., "An optimal control approach to a posteriori error estimation in finite element methods," *Acta Numerica 2000*, 2001, pp. 1–102.
- ²Giles, M. B. and Pierce, N. A., "Adjoint error correction for integral outputs," *Error Estimation and Adaptive Discretization Methods in Computational Fluid Dynamics*, edited by T. Barth and H. Deconinck, Vol. 25 of *Lecture Notes in Computational Science and Engineering*, Springer-Verlag, 2002.
- ³Barth, T., "Numerical Methods and Error Estimation for Conservation Laws on Structured and Unstructured Meshes," Lecture notes, von Karman Institute for Fluid Dynamics, Series: 2003-04, Brussels, Belgium, March 2003.
- ⁴Fidkowski, K. J. and Darmofal, D. L., "Review of Output-Based Error Estimation and Mesh Adaptation in Computational Fluid Dynamics," *AIAA Journal*, Vol. 49, No. 4, April 2011, pp. 673–694.
- ⁵Lu, J. and Darmofal, D. L., "Adaptive precision methodology for flow optimization via discretization and iteration error control," *AIAA Paper 2004–1096*, Jan 2004.
- ⁶Rannacher, R., "Adaptive solution of PDE-constrained optimal control problems," *The 2nd International Conference of Scientific Computing and Partial Differential Equations (SCPDE05)*, The First East Asia SIAM Symposium, Hong Kong, Dec. 2005, <http://numerik.iwr.uni-heidelberg.de/~rannache/>.
- ⁷Alauzet, F., Mohammadi, B., and Pironneau, O., "Mesh Adaptivity and Optimal Shape Design for Aerospace," *Springer Optimization and Its Applications*, edited by M. Pardalos, Springer, Aug. 2011, <http://www.ann.jussieu.fr/pironneau/>.
- ⁸Jameson, A., "Aerodynamic Design via Control Theory," *Journal of Scientific Computing*, Vol. 3, 1988, pp. 233–260, Also ICASE report 88–64.
- ⁹Nemec, M. and Zingg, D. W., "Newton–Krylov Algorithm for Aerodynamic Design Using the Navier–Stokes Equations," *AIAA Journal*, Vol. 40, No. 6, 2002, pp. 1146–1154.
- ¹⁰Peter, J. E. V. and Dwight, R. P., "Numerical Sensitivity Analysis for Aerodynamic Optimization: A Survey of Approaches," *Computers & Fluids*, Vol. 39, 2010, pp. 373–391.
- ¹¹Pironneau, O. and Polak, E., "Consistent Approximations and Approximate Functions and Gradients In Optimal Control," *SIAM Journal on Optimization*, Vol. 41, No. 2, 2000, pp. 487–510.
- ¹²Ziems, J. C. and Ulbrich, S., "Adaptive Multilevel Inexact SQP-Methods for PDE-Constrained Optimization," *SIAM Journal on Optimization*, Vol. 21, 2011, pp. 1–40.
- ¹³Dadone, A. and Grossman, B., "Progressive optimization of inverse fluid dynamic design problems," *Computers and Fluids*, Vol. 29, No. 1, 2000, pp. 1–32.
- ¹⁴Hicken, J. E. and Alonso, J. J., "PDE-constrained optimization with error estimation and control," *AIAA Paper 2012-151*, Nashville, TN, January 2012, (50th AIAA Aerospace Sciences Meeting).
- ¹⁵Nemec, M. and Aftosmis, M. J., "Adjoint Error Estimation and Adaptive Refinement for Embedded-Boundary Cartesian Meshes," *AIAA Paper 2007-4187*, Miami, FL, June 2007.
- ¹⁶Nemec, M., Aftosmis, M. J., and Wintzer, M., "Adjoint-Based Adaptive Mesh Refinement for Complex Geometries," *AIAA Paper 2008-0725*, Reno, NV, Jan. 2008.
- ¹⁷Aftosmis, M. J. and Nemec, M., "Exploring Discretization Error in Simulation-Based Aerodynamic Databases," *Proceedings of the 21st International Conference on Parallel Computational Fluid Dynamics*, edited by R. Biswas, May 2009.
- ¹⁸Nemec, M. and Aftosmis, M. J., "Adjoint Sensitivity Computations for an Embedded-Boundary Cartesian Mesh Method," *Journal of Computational Physics*, Vol. 227, 2008, pp. 2724–2742.

- ¹⁹Nemec, M. and Aftosmis, M. J., "Parallel Adjoint Framework for Aerodynamic Shape Optimization of Component-Based Geometry," AIAA Paper 2011-1249, Orlando, FL, Jan 2011, (49th AIAA Aerospace Sciences Meeting).
- ²⁰Aftosmis, M. J., Nemec, M., and Cliff, S. E., "Adjoint-Based Low-Boom Design with Cart3D," AIAA Paper 2011-3500, Honolulu, HI, June 2011, (29th AIAA Applied Aerodynamics Conference).
- ²¹Wintzer, M. and Kroo, I., "Optimization and Adjoint-Based CFD for the Conceptual Design of Low Sonic Boom Aircraft," AIAA Paper 2012-0963, Nashville, TN, January 2012, (50th AIAA Aerospace Sciences Meeting).
- ²²Nocedal, J. and Wright, S. J., *Nonlinear Optimization*, Springer, 2nd ed., 2006.
- ²³Lewis, A. S. and Overton, M. L., "Nonsmooth Optimization via BFGS," http://www.cs.nyu.edu/overton/papers/pdf/bfgs_inexactLS.pdf, 2008, Visited November 2012.
- ²⁴Lewis, A. S. and Overton, M. L., "Nonsmooth optimization via quasi-Newton methods," *Mathematical Programming*, Feb. 2012. doi:10.1007/s10107-012-0514-2.
- ²⁵Aftosmis, M. J., Berger, M. J., and Adomavicius, G., "A Parallel Multilevel Method for Adaptively Refined Cartesian Grids with Embedded Boundaries," AIAA Paper 2000-0808, Reno, NV, Jan. 2000.
- ²⁶Aftosmis, M. J., Berger, M. J., and Murman, S. M., "Applications of Space-Filling-Curves to Cartesian Methods for CFD," AIAA Paper 2004-1232, Reno, NV, Jan. 2004.
- ²⁷Berger, M. J., Aftosmis, M. J., and Murman, S. M., "Analysis of Slope Limiters on Irregular Grids," AIAA Paper 2005-0490, Reno, NV, Jan. 2005.
- ²⁸Venditti, D. A. and Darmofal, D. L., "Grid Adaptation for Functional Outputs: Application to Two-Dimensional Inviscid Flow," *Journal of Computational Physics*, Vol. 176, 2002, pp. 40–69.
- ²⁹Nemec, M., Aftosmis, M. J., Murman, S. M., and Pulliam, T. H., "Adjoint Formulation for an Embedded-Boundary Cartesian Method," AIAA Paper 2005-0877, Reno, NV, Jan. 2005.
- ³⁰Wintzer, M., Nemec, M., and Aftosmis, M. J., "Adjoint-Based Adaptive Mesh Refinement for Sonic Boom Prediction," *26th AIAA Applied Aerodynamics Conference*, No. 2008-6593, AIAA, Honolulu, HI, August 2008.
- ³¹Wintzer, M., "Span Efficiency Prediction Using Adjoint-Driven Mesh Refinement," *Journal of Aircraft*, Vol. 47, No. 4, July-August 2010, pp. 1468–1471.
- ³²Bakhtian, N. and Aftosmis, M. J., "Analysis of Inviscid Simulations for the Study of Supersonic Retropropulsion," AIAA Paper 2011-3194, Honolulu, HI, June 2011, (29th AIAA Applied Aerodynamics Conference).
- ³³Kless, J., Aftosmis, M. J., Ning, S. A., and Nemec, M., "Inviscid Analysis of Extended Formation Flight," *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, Big Island, Hawaii, 2012.
- ³⁴Lu, J., *An a posteriori Error Control Framework for Adaptive Precision Optimization using Discontinuous Galerkin Finite Element Method*, Ph.D. thesis, Massachusetts Institute of Technology, 2005.
- ³⁵Wintzer, M., Kroo, I., Aftosmis, M. J., and Nemec, M., "Conceptual Design of Low Sonic Boom Aircraft Using Adjoint-Based CFD," *Seventh International Conference on Computational Fluid Dynamics (ICCFD7)*, Big Island, Hawaii, 2012.
- ³⁶Rallabhandi, S. K., "Advanced Sonic Boom Prediction Using Augmented Burger's Equation," AIAA Paper 2011-1278, Orlando, FL, Jan. 2011, (49th AIAA Aerospace Sciences Meeting).